AdaBoost.C2: Boosting Classifiers Chains for Multi-label Classification

Jiaxuan Li,[1] Xiaoyan Zhu,[1,*] Jiayin Wang[1]

[1] School of Computer Science and Technology, Xi'an Jiaotong University, Xi'an, China
lijiaxuan@stu.xjtu.edu.cn, zhu.xy@xjtu.edu.cn, wangjiayin@mail.xjtu.edu.cn

Abstract

During the last decades, multi-label classification (MLC) has attracted the attention of more and more researchers due to its wide real-world applications. Many boosting methods for MLC have been proposed and achieved great successes. However, these methods only extend existing boosting frameworks to MLC and take loss functions in multi-label version to guide the iteration. These loss functions generally give a comprehensive evaluation on the label set entirety, and thus the characteristics of different labels are ignored. In this paper, we propose a multi-path AdaBoost framework specific to MLC, where each boosting path is established for distinct label and the combination of them is able to provide a maximum optimization to Hamming Loss. In each iteration, classifiers chain is taken as the base classifier to strengthen the connection between multiple AdaBoost paths and exploit the label correlation. Extensive experiments demonstrate the effectiveness of the proposed method.

Introduction

Multi-label classification (MLC) deals with the problem where each instance could belong to multiple labels simultaneously. In recent decades, quantities of algorithms have been proposed to address this problem. The most straightforward MLC method is Binary Relevance (BR) (Boutell et al. 2004), where the multi-label problem is partitioned into multiple single-label problems and each of them can be addressed by a binary classification algorithm. BR has the advantages of simplicity and extensibility; however, it never considers the correlation between labels and has a limited performance.

Adaptive Boosting (AdaBoost) (Freund and Schapire 1997) is an effective approach to iteratively combine a committee of weak hypotheses to a strong one, and has been well received in various fields (Ferreira and Figueiredo 2012). AdaBoost.MH (Schapire and Singer 2000) first extends AdaBoost to MLC, where BR is taken to induce base classifiers and Hamming Loss is employed as the loss function. Then, such scheme is widely used and improved by many algorithms (Johnson and Cipolla 2005; Wang et al. 2019; Zhang and Jung 2020; Zhang, Ramaswamy, and Agarwal 2020); however, their loss functions are decomposable to labels and neglect the label correlation (Dembczyński et al. 2012). To address this issue, novel frameworks available to nondecomposable loss function are further proposed (Amit, Dekel, and Singer 2007; Rapp et al. 2020). And some researchers (Dembczynski, Kotlowski, and Hüllermeier 2012; Jung and Tewari 2018) employ the ranking relation between labels to construct loss function. Nonetheless, the loss functions in aforementioned works only measure the comprehensive performance of a learner on all labels, so the multiple single-label learners contained in each base classifier always share same weight, which fails to consider the different abilities of a base classifier for different labels.

Classifiers Chain (CC) (Read et al. 2009) takes another approach to improve BR. In CC, a chain of classifiers is constructed for different labels, where the prediction information of preceding classifiers can be utilized by subsequent ones and thus the label correlation is considered. However, CC incorporates the error propagation problem (Senge, Coz, and Hüllermeier 2014), that is, the errors produced in the front may propagate down the chain and cause further errors. Many methods propose to address this problem by heuristically providing a good chain order (Kajdanowicz and Kazienko 2013; Jun et al. 2019). As an optimal order is hard to find, some ensemble methods (Read et al. 2011; Trajdos and Kurzynski 2019), instead, provide an ensemble of CC learners in different orders to avoid the uncertainty of a single order. Unfortunately, so far, the error propagation problem, together with the chain order issue, have not been well solved (Read et al. 2021).

In this paper, we propose a novel method named AdaBoost.C2, which absorbs the advantages of AdaBoost and CC, and further eliminates the defects of the two methods. First, a multi-path AdaBoost framework is proposed to fully consider the difference between multiple labels. For each label, a succession of single-label learners is organized into a boosting classifier, and the loss of each single-label learner is taken to determine its weight as well as the data distribution in next iteration. Since multiple boosting processes are asynchronously moved and stopped, each boosting classifier is able to fit distinct label in the maximum extent, and the time complexity of the multi-label model is reduced. Second, in each iteration, CC method is taken to construct base classifiers, which effectively considers the correlation

[*]indicates the corresponding author.

between labels. To address the error propagation issue, the chain orders of CC classifiers are heuristically determined by the loss function values of AdaBoost algorithms. As CC classifiers can be induced in different orders, the diversity of base classifiers is further enhanced. To summarize, the contributions of AdaBoost.C2 can be highlighted as follows:

- AdaBoost and CC are combined to tackle MLC problem, where boosting algorithm effectively enhances the model accuracy and CC classifiers exploit the label correlation.
- A multi-path AdaBoost framework is presented to learn the difference between multiple labels, which realizes an effective multi-label model and reduces the time complexity simultaneously.
- The loss function values produced by AdaBoost are taken to heuristically guide the chain order of CC, which effectively alleviates the error propagation problem and improves the diversity of base classifiers.

Existing boosting methods for MLC

Many boosting methods have been proposed to tackle MLC problem. Especially, as one of the most common boosting scheme, AdaBoost (Freund and Schapire 1997) and its its variants, e.g. GBDT, XGBoost, have also been widely applied. To cope with the increased dimension of label space, these methods make improvements in mainly two aspects, i.e. the design of loss function and base classifier.

Loss function

The loss functions for MLC can be categorized according to whether they are decomposable to labels. Decomposable metrics like Hamming Loss measure the average performance of a learner on all labels, which is convenient to adapt to various multi-label algorithms. Nondecomposable metrics, instead, give an overall evaluation on multiple labels. For example, Subset Accuracy evaluates whether a learner correctly predicts all the labels of an instance.

AdaBoost.MH (Schapire and Singer 2000) first takes Hamming Loss as the surrogate loss function to extend AdaBoost to MLC, and such technique is widely used in many MLC tasks (Johnson and Cipolla 2005; Wang et al. 2019). To consider the interaction between labels, (Amit, Dekel, and Singer 2007) gives the differentiable upper boundary of Subset Accuracy and incorporates it in MLC. BOOMER (Rapp et al. 2020, 2021) further establishes a multi-label boosting framework available to more multi-label loss functions. Nonetheless, Subset Accuracy takes only $\{0, 1\}$ for each unknown instance and has a poor discriminate ability. To flexibly capture label dependency, some ranking-based methods (Dembczynski, Kotlowski, and Hüllermeier 2012; Jung and Tewari 2018) take the pairwise ranking relation between positive and negative labels to guide the iteration.

Base classifier

Basic MLC methods can be summarized to mainly two types (Gibaja and Ventura 2015): Algorithm Adaptation (AA) and Problem Transformation (PT). AA adapts single-label algorithms to their multi-label versions (Zhang and Zhou 2007; Zhang and Zhang 2010) and usually tails for special tasks, e.g. multi-label image classification (MLIC), MLC with false positive labels (partial multi-label learning, PML), MLC with class imbalance (MLC.IM), MLC with high dimensional sparse label space (extreme multi-label learning, XML) and multi-label ranking (MLR). Conversely, PT transforms MLC problem to well-established single-label problems and has a wider range of applications. PT methods include Binary Relevance (BR) (Boutell et al. 2004), Classifiers Chain (CC) (Read et al. 2009) and Label Powerset (LP) (Tsoumakas and Vlahavas 2007). In BR and CC, the original problem is partitioned into multiple binary classification problems and solved one by one, while LP transforms the multi-label problem to a multi-class classification problem and gives the predictions for multiple labels altogether.

In summary, for loss function, decomposable metrics are widely used in boosting methods for MLC. For base classifier, AA is usually taken for special tasks and BR is widely employed otherwise. And some representative methods are listed in the Tab. 1.

ID	Loss function	Base	Region
1	Decomposable	BR	MLC
2	Decomposable	AA	MLIC
3	Decomposable	BR	MLC
4	Decomposable	BR	PML
5	Decomposable	AA	XML
6	Decomposable	AA	MLC.IM
7	Decomposable	CC	MLC
8	Decomposable	AA	XML
9	Subset Accuracy	BR	MLC
10	Subset Accuracy	AA	MLC
11	Ranking-based	AA	MLR
12	Ranking-based	AA	MLR

Table 1: Some representative boosting MLC algorithms. Notes: Base for base classifier, Region for application region, 1 for (Freund and Schapire 1997), 2 for (Johnson and Cipolla 2005), 3 for (Al-Salemi, Noah, and Ab Aziz 2016), 4 for (Wang et al. 2019), 5 for (Si et al. 2017), 6 for (Cheng et al. 2020), 7 for (Bohlender, Loza Mencía, and Kulessa 2020), 8 for (Zhang and Jung 2020), 9 for (Amit, Dekel, and Singer 2007), 10 for (Rapp et al. 2021), 11 for (Dembczynski, Kotlowski, and Hüllermeier 2012), 12 for (Jung and Tewari 2018).

AdaBoost.C2

General view

To make a clear explanation, the descriptions to all symbols are given in Tab. 2.

As stated in previous section, it is convenient to establish a common MLC framework with decomposable loss function. As the most representative loss function in this scheme, the optimization objective of Hamming Loss can be defined as:

$$J = \min HL = \min \frac{1}{N}\frac{1}{Q}\sum_{n=1}^{N}\sum_{q=1}^{Q}\mathbb{I}(Y_{nq} \neq \hat{Y}_{nq}) \qquad (1)$$

Notation	Description
$\mathcal{X} = \mathbb{R}^d$	Feature space
$\mathcal{Y} = \{Y_q \mid 1 \le q \le Q\}$	Label space
$\mathcal{X}^q = \mathcal{X} \times Y_1 \times \cdots \times Y_q$	Extended space of $\mathcal{X}$
$\mathcal{D} = \{w_n \mid 1 \le n \le N\}$	Distribution of data
$C = \{C_t \mid 1 \le t \le T\}$	Ensemble model
$C_{tq} = \mathcal{L}(\mathcal{X}, Y_q)$	Single-label learner
$C_t = \{C_{tq} \mid 1 \le q \le Q\}$	t-th base classifier
$C_{\sim q} = \sum_{t=1}^{T} \alpha_{tq} C_{tq}$	q-th boosting classifier
$\ell(C_{tq} \mid \mathcal{D}) = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[\ell(C_{tq})]$	Loss of C_{tq}
$O_t = (Y_{(q)} \mid 1 \le q \le Q)$	Chain order of C_t
$\boldsymbol{y}_m = \{C_{tq}(\boldsymbol{x}_m) \mid 1 \le q \le Q\}$	Inference of $\boldsymbol{x}_m$

Table 2: Main notations.

where HL is the Hamming Loss and $\mathbb{I}(\cdot)$ is the indicator function.

According to the inequality $\sum_i \min f_i \le \min \sum_i f_i$, $f_i \ge 0$, a lower bound of loss function based on Hamming Loss in Eq. 1 can be reformulated as:

$$J = \frac{1}{Q} \sum_{q=1}^{Q} \min HL_q \le \min HL \tag{2}$$

where $HL_q = 1/N \sum_{n=1}^{N} \mathbb{I}(Y_{nq} \ne \hat{Y}_{nq})$ is the Hamming Loss corresponding to the q-th label.

That is, theoretically, it is better to separately optimize Q labels than to optimize all labels simultaneously. On basis of this analysis, we take multiple AdaBoost processes to construct a multi-label model, where each boosting path aims to minimize the Hamming Loss of the corresponding label. Undoubtedly, the proposed multi-path AdaBoost framework fully takes into account the difference between multiple labels. To further consider the label correlation, we employ CC to build base classifiers, and the proposed method is so-called AdaBoost.C2. Moreover, the chain order of each CC classifier is heuristically determined by the loss function values of AdaBoost processes, which effectively eliminates the uncertainty brought by error propagation and increases the diversity of base classifiers. In summary, there are two kinds of paths of basic single-label learner induction in AdaBoost.C2: 1) Q AdaBoost paths, and 2) T CC paths (chain orders). The diagrammatic sketch of AdaBoost.C2 is depicted as Fig. 1. And the details of model induction will be described in the following.

Multi-path AdaBoost framework

The main difference between previous boosting MLC methods and AdaBoost.C2 is the way boosting method applied, as depicted in Fig. 2. In the conventional scheme, the iteration is based on the performance of each base classifier, so that all the single-label learners contained within a base classifier always share same weight and trained in equally distributed dataset. In AdaBoost.C2, multiple AdaBoost paths are established, so multiple single-label learners in a base classifier are able to obtain different weights and trained in specially distributed datasets.

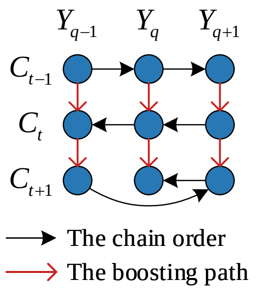

Figure 1: The local structure of AdaBoost.C2, where the boosting path is predetermined, and the chain orders of base classifiers are determined in the training process.

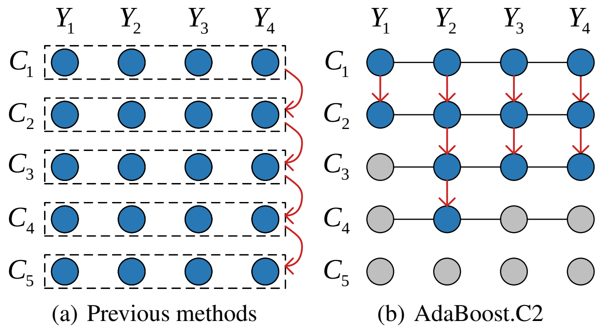

Figure 2: Toy example: Two types of boosting schemes for MLC. In previous methods, the boosted base classifier is viewed as a entirety. In AdaBoost.C2, multiple boosting paths are constructed specific to different labels, and each of them can be asynchronously induced and stopped.

As different labels are equally treated, we take the q-th label as an example for a clear description. For Y_q, a committee of T single-label learners is induced to form the q-th boosting classifier. In AdaBoost.C2, the exponential loss, a upper boundary of Hamming Loss, is taken as the surrogate loss function:

$$\ell = \ell(C_{\sim q} \mid \mathcal{D}) = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[e^{-C_{\sim q}(\boldsymbol{x}) y_q}] \tag{3}$$

Specifically, the loss function for single-label learner C_{tq} can be simplified as:

$$\ell = \ell(\alpha_{tq} C_{tq} \mid \mathcal{D}) = e^{-\alpha_{tq}}(1 - \epsilon_{tq}) + e^{\alpha_{tq}} \epsilon_{tq} \tag{4}$$

where α_{tq} is the weight of C_{tq} and $\epsilon_{tq} = P_{x\sim\mathcal{D}}(C_{tq}(x) \ne y_q)$ denotes the error rate.

Minimize Eq. 4, and the weight of this basic single-label learner can be obtained:

$$\alpha_{tq} = \frac{1}{2} \ln \frac{1 - \epsilon_{tq}}{\epsilon_{tq}} \tag{5}$$

And the data distribution can also be accordingly updated:

$$\mathcal{D}_{t+1,q} = \frac{e^{-y_q \alpha_{tq} C_{tq}(x)}}{Z_{tq}} \mathcal{D}_{t,q} \tag{6}$$

where the Z_{tq} is a smooth parameter to ensure that $\mathcal{D}_{t+1,q}$ meets the characteristics of a distribution.

Eventually, the q-th boosting classifier can be obtained:

$$C_{\sim q} = \sum_{t=1}^{T} \alpha_{tq} C_{tq} \tag{7}$$

Another difference between existing boosting MLC methods and AdaBoost.C2 is that AdaBoost.C2 tends to stop the iteration in advance. In conventional scheme, the loss function is on the top of a Q-dimensional label space and achieves maximum optimization when all labels match the optimal solution synchronously. On the contrary, AdaBoost.C2 allows multiple labels to be optimized asynchronously. That is, the loss function of AdaBoost.C2 is on the top of a 1-dimensional label space, so each boosting classifier is easier to be better optimized and early terminated. Therefore, there may be great difference in the number of the iteration rounds for different labels. To equally treat labels with boosting classifiers in different rounds, we normalize the weights of all basic single-label learners within a boosting classifier, and Eq. 7 is rewritten as:

$$C_{\sim q} = \sum_{t=1}^{T'} \frac{\alpha_{tq}}{\alpha_{\sim q}} C_{tq} \tag{8}$$

where $T' \leq T$ is the number of iteration rounds in reality, and $\alpha_{\sim q} = \sum_{t=1}^{T'} \alpha_{tq}$.

Base CC classifier

In the multi-path AdaBoost framework, multiple boosting classifiers are independently induced, which focuses on the difference between labels but neglects the correlation between them. To address this issue, we take the methodology of Classifier Chain (CC) to construct base classifiers. That is, in each iteration, a chain of classifiers is orderly induced, where the outputs of preceding classifiers can be utilized by subsequent ones. In AdaBoost.C2, the chain order is heuristically guided by the loss function values. To alleviate the error propagation, single-label learners with lower errors are put in the front of chain and the errors of current CC classifier are approximated by the previous iteration. For example, the chain order of C_t is determined by the loss function values of C_{t-1}:

$$O_t = argsort(\epsilon_{t-1,1}, \epsilon_{t-1,2}, \cdots, \epsilon_{t-1,Q}) \tag{9}$$

where $argsort(\cdot)$ denotes that sort from small to large and return the indices.

Specifically, the induction process of C_t can be formulated as:

$$\begin{aligned} & C_{t(1)} = \mathcal{L}(\mathcal{X}, Y_{(1)}) \\ & C_{t(2)} = \mathcal{L}(\mathcal{X}^1, Y_{(2)}), \mathcal{X}^1 = \mathcal{X} \times Y_{(1)} \\ & \quad \cdots \\ & C_{t(Q)} = \mathcal{L}(\mathcal{X}^{Q-1}, Y_{(Q)}), \mathcal{X}^{Q-1} = \mathcal{X}^{Q-2} \times Y_{(Q-1)} \end{aligned} \tag{10}$$

And the inference of base classifier C_t given an unknown instance $\boldsymbol{x}_m$ can be accordingly expressed as:

$$\begin{aligned} & y_{m(1)} = C_{t(1)}(\boldsymbol{x}_m) \\ & y_{m(2)} = C_{t(2)}(\boldsymbol{x}_m^1), \boldsymbol{x}_m^1 = \boldsymbol{x}_m \cup y_{m(1)} \\ & \quad \cdots \\ & y_{m(Q)} = C_{t(Q)}(\boldsymbol{x}_m^{Q-1}), \boldsymbol{x}_m^{Q-1} = \boldsymbol{x}_m^{Q-2} \cup y_{m(Q-1)} \end{aligned} \tag{11}$$

In empirical practice, the boosting sequence for some labels may be early terminated as shown in Fig. 2(b). A possible occurrence of chain order corresponding to this toy example is shown in Fig. 3:

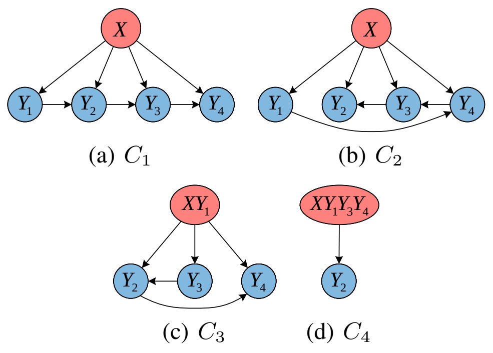

Figure 3: Chain orders corresponding to the toy example in Fig. 2(b), where 4 boosting classifiers stop their iterations after 2, 4, 3, 3 rounds, respectively.

In such situation, the induction of base classifiers should be accordingly adjusted. Take C_3 for example. According to the chain order $O_3 = (Y_{(1)}, Y_{(2)}, \cdots, Y_{(Q)}) = (\phi, Y_3, Y_2, Y_4)$, the induction of C_3 should be redefined as:

$$\begin{aligned} & C_{31} = C_{21} \\ & C_{33} = \mathcal{L}(\mathcal{X}^1, Y_3), \mathcal{X}^1 = \mathcal{X} \times Y_1 \\ & C_{32} = \mathcal{L}(\mathcal{X}^2, Y_2), \mathcal{X}^2 = \mathcal{X}^1 \times Y_3 \\ & C_{34} = \mathcal{L}(\mathcal{X}^3, Y_4), \mathcal{X}^3 = \mathcal{X}^2 \times Y_2 \end{aligned} \tag{12}$$

where C_{21} has obtained an ideal result and the boosting sequence for Y_1 has been stopped, so C_{31} can be a duplicate of C_{21}. Finally, this base classifier can be obtained, $C_3 = \{C_{21}, C_{32}, C_{33}, C_{34}\}$, and its inference for $\boldsymbol{x}_m$ can be obtained according to Eq. 11.

Experiments

Experimental setting

Datasets. In the experiments, 26 datasets from 7 different domains are utilized. All these datasets are downloaded from the website of KDIS [1]. Details about the datasets are shown in Table 3, including number of instances (N), features (D) and labels (Q) of the datasets.

Comparison methods. To report reliable comparison results, MLDE is compared with 4 state-of-the-art MLC ensemble methods proposed in recent years, including DECC (Trajdos and Kurzynski 2019), ACkEL (Wang et al. 2021), MLWSE (Xia, Chen, and Yang 2021), BOOMER (Rapp et al. 2020, 2021). DECC heuristically generates CC base classifiers to learn label correlation, and employs dynamic ensemble method to combine them. ACkEL takes the framework of RAkEL (Tsoumakas and Vlahavas 2007) (one of the most representative LP methods) to retain label correlation

[1]http://www.uco.es/kdis/mllresources/

ID	Dataset	Domain	N	D	Q	ID	Dataset	Domain	N	D	Q
#1	3Sources_bbc	Text	352	1000	6	#14	Langlog	Text	1460	1004	75
#2	3Sources_guardian	Text	302	1000	6	#15	Medical	Text	978	1449	45
#3	3Sources_inter	Text	169	3000	6	#16	PlantGO	Biology	978	3091	12
#4	3Sources_reuters	Text	294	1000	6	#17	Scene	Image	2407	294	6
#5	Birds	Audio	645	260	19	#18	Slashdot	Text	3782	1079	22
#6	CAL500	Music	502	68	174	#19	Chemistry	Text	6961	540	175
#7	CHD_49	Medicine	555	49	6	#20	Chess	Text	1675	585	227
#8	Enron	Text	1702	1001	53	#21	Coffee	Text	225	1763	123
#9	Flags	Image	194	19	7	#22	VirusGO	Biology	207	749	6
#10	Foodtruck	Recommend	407	21	12	#23	Yeast	Biology	2417	103	14
#11	GnegativeGO	Biology	1392	1717	8	#24	Yelp	Text	10810	671	5
#12	GpositiveGO	Biology	519	912	4	#25	Corel5k	Image	5000	499	374
#13	Image	Image	2000	294	5	#26	Philosophy	Text	3971	842	233

Table 3: Benchmark datasets.

information, and takes active learning approach to enhance model accuracy. MLWSE introduces a weighted ensemble model with an optimization objective, where graph-based label correlation constraints and regularization terms are incorporated. BOOMER establishes a MLC gradient boosting framework available to nondecomposable loss function and designs a tree-based AA classifier to further exploit label dependency.

Hyper-parameter setting. For AdaBoost.C2, we set $T = 10$ as its upper limit of iteration rounds and $\delta = 0.01$ as the lower limit of iteration error. For comparison methods, we take the recommended hyper-parameters in corresponding literatures (e.g. $k = 10$ for kNN in DECC). And for basic learner, if it is not specified, RBF kerneled SVM with defaults recommended in scikit-learn is used. All codes are openly available [2].

Comparison results

We compared the proposed AdaBoost.C2 with the 4 baseline methods on 26 benchmark datasets. For each dataset, we conduct 10 fold validation test for 5 times and calculate the average values. The comparison in terms of 4 well-known evaluation metrics in MLC (Hamming Loss, Coverage, Ranking Loss, Average Precision) (Madjarov et al. 2012) is reported in Tab. 4.

Primarily, according to the experimental results, the weighted ensemble methods (trainable weighted method: MLWSE, and boosting methods: BOOMER, AdaBoost.C2) perform much better than those of nonweighted methods (DECC, ACkEL), which demonstrates the effectiveness of boosting approach. Secondly, among the total 104 cases over 26 benchmark datasets and 4 evaluation metrics, AdaBoost.C2 ranks first in 65 cases and outperforms the comparison methods, greatly surpassing other weighted methods (21 for MLWSE, 17 for BOOMER). Specifically, AdaBoost.C2 outperforms DECC, ACkEL, MLWSE, and BOOMER in 100, 102, 73 and 82 cases respectively and obtains highest average rank in all metrics. Especially, in the ranking-based metrics specific to MLC tasks (Coverage, Ranking Loss and Average Precision), AdaBoost.C2 ranks first in 50 cases out of total 78 ones, which demonstrates the strength of the proposed method in label correlation exploration. Thirdly, compare to MLWSE, the most competent comparison algorithm, AdaBoost.C2 achieves an average improvement of 2.63%, 10.66%, 12.56% and 4.00% in 4 evaluation metrics, respectively.

Efficiency analysis

In AdaBoost.C2, both multi-path AdaBoost framework and heuristically trained CC classifiers are time consuming. However, the multiple AdaBoost paths are asynchronously trained and the each AdaBoost path is more easier to be stopped in advance comparing to other boosting methods for MLC. And the chain orders of CC classifiers are guided by the loss function values produced by boosting processes, which never incorporates extra computational complexity. To verify the model efficiency, we compare the running time of AdaBoost.C2 with 2 boosting-based baselines, AdaBoost.MH and BOOMER. AdaBoost.MH applies conventional boosting framework to MLC, and BOOMER further considers the correlation information in label space. The runtime comparison results are shown in Fig. 4, and all the methods are executed on Intel(R) Core(TM) i5-9500 CPU @ 3.00GHz. In addition, the comparison on iteration rounds between AdaBoost.C2 and AdaBoost.MH is shown in Tab. 5.

From the runtime comparison results, AdaBoost.C2 exceeds two baselines in 15 and 14 datasets out of 26 ones, respectively. And AdaBoost.C2 achieves the best performance on the average running time of all datasets, especially it leads significantly in some large label scale datasets. And the iteration rounds comparison results are also consistent with our analysis. To summarize, AdaBoost.C2 has a competitive efficiency in both theory and practice.

Ablation study

In this paper, multi-path AdaBoost framework and base CC classifiers are combined to construct AdaBoost.C2. Specifically, the multi-path AdaBoost framework is taken to learn

[2]https://github.com/JiaxuanGood/AdaBoostC2.git

	Hamming Loss ↓					Coverage ↓				
Dataset	DECC	ACkEL	MLWSE	BOOMER	AdaC2	DECC	ACkEL	MLWSE	BOOMER	AdaC2
#1	0.296	0.409	**0.196**	0.249	0.200	0.412	0.592	0.352	0.372	**0.344**
#2	0.284	0.367	0.200	0.263	**0.193**	0.391	0.587	**0.387**	0.399	0.395
#3	0.297	0.381	0.222	0.280	**0.201**	0.435	0.587	0.406	0.453	**0.370**
#4	0.288	0.436	0.200	0.258	**0.190**	0.380	0.598	0.382	0.382	**0.380**
#5	0.058	0.347	0.046	**0.021**	0.053	0.221	0.328	0.133	**0.104**	0.187
#6	0.154	0.414	0.165	0.161	**0.139**	0.850	0.938	0.902	0.870	**0.779**
#7	0.314	0.474	0.298	0.297	**0.284**	0.485	0.863	0.458	0.515	**0.437**
#8	0.044	0.360	0.039	0.041	**0.034**	0.366	0.517	0.277	0.473	**0.209**
#9	0.328	0.386	0.260	**0.245**	0.321	0.568	0.816	0.543	**0.533**	0.541
#10	0.169	0.462	0.158	**0.153**	0.156	0.336	0.565	0.323	0.421	**0.302**
#11	0.020	0.278	**0.013**	0.013	0.014	0.025	0.379	0.015	0.024	**0.014**
#12	0.036	0.344	0.032	**0.031**	0.031	0.032	0.583	**0.023**	0.034	0.024
#13	0.177	0.411	0.155	**0.143**	0.163	0.213	0.635	**0.171**	0.224	0.186
#14	0.174	0.533	0.162	0.165	**0.149**	0.764	0.920	0.690	0.774	**0.639**
#15	0.016	0.224	0.011	**0.010**	0.010	0.108	0.504	0.047	0.151	**0.034**
#16	0.057	0.339	0.037	**0.033**	0.038	0.092	0.554	0.044	0.106	**0.043**
#17	0.095	0.492	0.077	**0.070**	0.076	0.100	0.569	0.067	0.146	**0.063**
#18	0.030	0.210	0.016	0.016	**0.015**	0.123	0.278	0.046	0.109	**0.035**
#19	0.012	**0.007**	0.012	0.013	0.011	0.457	0.606	0.276	0.556	0.180
#20	0.010	0.335	0.010	0.010	**0.010**	0.505	0.676	0.338	0.629	**0.197**
#21	0.016	0.264	0.024	0.017	**0.015**	0.415	0.530	0.412	0.469	**0.268**
#22	0.065	0.415	0.050	**0.042**	0.052	0.087	0.621	**0.070**	0.099	0.071
#23	0.208	0.481	**0.190**	0.192	0.194	0.464	0.847	0.450	**0.446**	0.451
#24	0.230	0.544	0.132	0.153	**0.122**	0.248	0.606	0.192	0.240	**0.186**
#25	0.010	0.355	**0.009**	0.010	0.010	0.422	0.554	0.465	0.452	**0.288**
#26	0.010	0.317	0.009	0.009	**0.009**	0.479	0.619	0.297	0.582	**0.180**
Rank	3.81	4.85	2.31	2.23	**1.81**	3.12	5.00	2.08	3.42	**1.38**
	Ranking Loss ↓					Average Precision ↑				
#1	0.461	0.728	0.393	0.410	**0.382**	0.433	0.440	**0.519**	0.516	0.504
#2	0.439	0.727	**0.432**	0.437	0.441	0.469	0.449	0.480	**0.487**	0.443
#3	0.482	0.728	0.452	0.496	**0.409**	0.430	0.456	0.462	0.446	**0.469**
#4	0.425	0.731	0.425	**0.419**	0.419	0.487	0.437	0.493	**0.502**	0.469
#5	0.190	0.402	**0.103**	0.117	0.155	0.650	0.601	0.785	**0.808**	0.676
#6	0.231	0.776	0.262	0.289	**0.186**	0.439	0.237	0.422	0.363	**0.493**
#7	0.255	0.787	0.215	0.289	**0.198**	0.754	0.594	0.788	0.748	**0.805**
#8	0.223	0.504	0.133	0.305	**0.090**	0.610	0.431	0.659	0.512	**0.712**
#9	0.241	0.758	0.217	**0.215**	0.216	0.796	0.688	**0.813**	0.808	0.811
#10	0.181	0.530	0.169	0.237	**0.153**	0.716	0.504	0.732	0.703	**0.743**
#11	0.020	0.499	0.010	0.018	**0.009**	0.961	0.568	**0.977**	0.969	0.975
#12	0.040	0.640	**0.028**	0.042	0.029	0.958	0.571	**0.967**	0.962	0.966
#13	0.197	0.675	**0.146**	0.197	0.163	0.775	0.515	**0.823**	0.801	0.802
#14	0.328	0.657	0.211	0.357	**0.177**	0.519	0.298	0.618	0.494	**0.657**
#15	0.079	0.634	0.031	0.110	**0.021**	0.784	0.195	0.876	0.818	**0.901**
#16	0.089	0.717	0.038	0.099	**0.038**	0.798	0.299	0.888	0.850	**0.890**
#17	0.100	0.674	0.063	0.148	**0.059**	0.845	0.447	0.886	0.853	**0.890**
#18	0.107	0.391	0.037	0.088	**0.028**	0.803	0.548	0.919	0.853	**0.926**
#19	0.297	0.886	0.156	0.388	0.100	0.260	0.158	0.398	0.271	0.421
#20	0.307	0.693	0.175	0.436	**0.094**	0.300	0.045	0.425	0.215	**0.492**
#21	0.280	0.710	0.266	0.322	**0.153**	0.247	0.064	0.324	0.289	**0.511**
#22	0.054	0.695	**0.037**	0.067	0.039	0.923	0.428	**0.941**	0.931	0.937
#23	0.187	0.765	**0.165**	0.193	0.169	0.742	0.465	**0.772**	0.740	0.763
#24	0.149	0.523	0.073	0.126	**0.066**	0.859	0.689	0.927	0.886	**0.936**
#25	0.215	0.742	0.206	0.233	**0.122**	0.181	0.100	0.295	0.210	**0.318**
#26	0.309	0.769	0.156	0.423	**0.089**	0.282	0.033	0.451	0.214	**0.493**
Rank	3.27	5.00	1.88	3.42	**1.42**	3.65	4.85	1.77	3.08	**1.65**

Table 4: Comparison results between AdaBoost.C2 (abbreviated as AdaC2) and other methods. The Rank represents the average rank of an algorithm on all datasets.

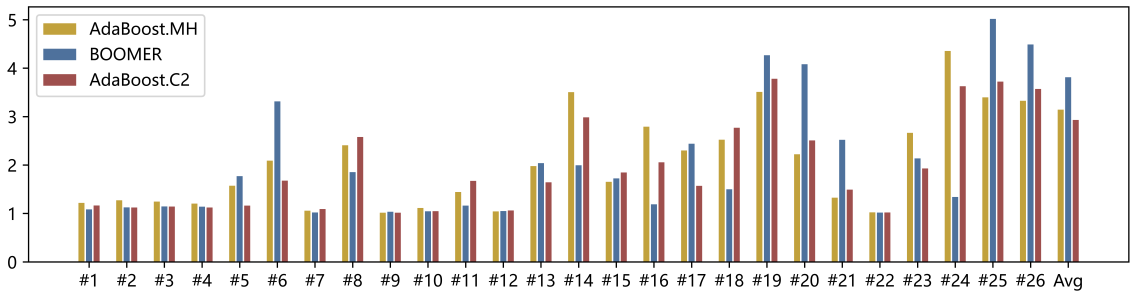

Figure 4: Running time comparison. The height of one column is $h = \log(10 + t)$, where t is the runtime in seconds.

DataID	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10	#11	#12	#13
AdaBoost.C2	3.20	3.18	3.17	3.17	2.14	2.84	3.80	1.69	4.54	2.90	1.13	1.52	3.83
AdaBoost.MH	2.8	5.5	4.1	3.3	5.1	6.7	1	1	3.4	4.3	1	2.8	4.1
DataID	#14	#15	#16	#17	#18	#19	#20	#21	#22	#23	#24	#25	#26
AdaBoost.C2	2.73	1.17	1.28	2.65	1.27	1.27	1.20	1.36	1.72	3.16	2.46	1.17	1.15
AdaBoost.MH	10	1	10	10	1	1	1	1	1	10	10	1	1

Table 5: Average number of iteration rounds comparison.

the difference between labels, while CC classifiers are employed to exploit the correlation between labels. To verify their contributions, an ablation study is conducted by comparing AdaBoost.C2 with 2 baselines: 1) Base1: A combination of BR and conventional AdaBoost scheme, neither label difference nor label correlation is considered; 2) Base2: A combination of BR and multi-path AdaBoost scheme, considering only the label difference. Literally, Base1 is equivalent to the first boosting MLC method AdaBoost.MH (Schapire and Singer 2000). The ablation study results are depicted as the box charts in Fig. 5.

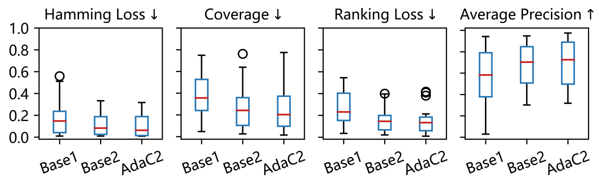

Figure 5: Ablation study.

It can be seen that both Base2 and AdaBoost.C2 have a good performance in terms of Hamming Loss, which implies the successful optimization of multi-path AdaBoost framework to Hamming Loss. And in other 3 metrics, the ranking of algorithms is $AdaBoost.C2 \succ Base2 \succ Base1$. That is, the combination of multi-path AdaBoost framework and heuristically ordered base CC classifiers gains the excepted promotion, and AdaBoost.C2 is able to learn the correlation and difference between labels simultaneously.

Parameter sensitivity analysis

In AdaBoost.C2, the termination condition of iteration is determined by the lower limit of boosting error δ and the upper limit of iteration rounds T. We analyze the effect of δ on average performance and iteration rounds statistics of 26 datasets. According to the results shown in Fig. 6, we set $\delta = 0.01$, and $T = 10$ because AdaBoost.C2 stops its iteration within 10 rounds in most datasets.

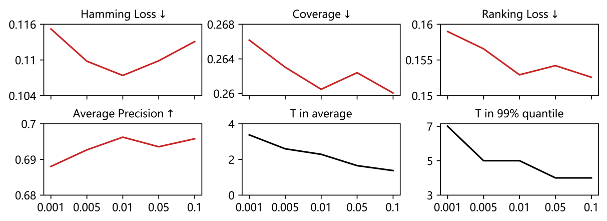

Figure 6: Parameter analysis.

Conclusions

In this paper, we propose a multi-path AdaBoost framework for MLC problem and provide a concrete instantiation named AdaBoost.C2. In AdaBoost.C2, multiple AdaBoost paths are constructed specific to different labels and taken to alleviate the error propagation problem in CC, and CC classifiers are generated to enhance the interaction between AdaBoost paths. Extensive experiments demonstrated the effectiveness of AdaBoost.C2.

AdaBoost.C2 also provides a novel pattern to tackle MLC problem. In recent studies, the exploration on label correlation is generally put into the first place. The effectiveness of AdaBoost.C2 implies that the difference between labels is also of great significance and is easy to be neglected due to the over consideration on label correlation.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No.92046009, 72274152, 71872146).

References

Al-Salemi, B.; Noah, S. A. M.; and Ab Aziz, M. J. 2016. RFBoost: an improved multi-label boosting algorithm and its application to text categorisation. *Knowledge-Based Systems*, 103: 104–117.

Amit, Y.; Dekel, O.; and Singer, Y. 2007. A boosting algorithm for label covering in multilabel problems. In *Artificial Intelligence and Statistics*, 27–34. PMLR.

Bohlender, S.; Loza Mencía, E.; and Kulessa, M. 2020. Extreme gradient boosted multi-label trees for dynamic classifier chains. In *International Conference on Discovery Science*, 471–485. Springer.

Boutell, M. R.; Luo, J.; Shen, X.; and Brown, C. M. 2004. Learning multi-label scene classification. *Pattern recognition*, 37(9): 1757–1771.

Cheng, K.; Gao, S.; Dong, W.; Yang, X.; Wang, Q.; and Yu, H. 2020. Boosting label weighted extreme learning machine for classifying multi-label imbalanced data. *Neurocomputing*, 403: 360–370.

Dembczynski, K.; Kotlowski, W.; and Hüllermeier, E. 2012. Consistent multilabel ranking through univariate losses. *arXiv preprint arXiv:1206.6401*.

Dembczyński, K.; Waegeman, W.; Cheng, W.; and Hüllermeier, E. 2012. On label dependence and loss minimization in multi-label classification. *Machine Learning*, 88(1): 5–45.

Ferreira, A. J.; and Figueiredo, M. A. 2012. Boosting algorithms: A review of methods, theory, and applications. *Ensemble machine learning*, 35–85.

Freund, Y.; and Schapire, R. E. 1997. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1): 119–139.

Gibaja, E.; and Ventura, S. 2015. A tutorial on multilabel learning. *ACM Computing Surveys (CSUR)*, 47(3): 1–38.

Johnson, M.; and Cipolla, R. 2005. Improved Image Annotation and Labelling through Multi-Label Boosting. In *BMVC*.

Jun, X.; Lu, Y.; Lei, Z.; and Guolun, D. 2019. Conditional entropy based classifier chains for multi-label classification. *Neurocomputing*, 335: 185–194.

Jung, Y. H.; and Tewari, A. 2018. Online boosting algorithms for multi-label ranking. In *International Conference on Artificial Intelligence and Statistics*, 279–287. PMLR.

Kajdanowicz, T.; and Kazienko, P. 2013. Heuristic classifier chains for multi-label classification. In *International Conference on Flexible Query Answering Systems*, 555–566. Springer.

Madjarov, G.; Kocev, D.; Gjorgjevikj, D.; and Džeroski, S. 2012. An extensive experimental comparison of methods for multi-label learning. *Pattern recognition*, 45(9): 3084–3104.

Rapp, M.; Mencía, E. L.; Fürnkranz, J.; and Hüllermeier, E. 2021. Gradient-based label binning in multi-label classification. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 462–477. Springer.

Rapp, M.; Mencía, E. L.; Fürnkranz, J.; Nguyen, V.-L.; and Hüllermeier, E. 2020. Learning gradient boosted multi-label classification rules. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 124–140. Springer.

Read, J.; Pfahringer, B.; Holmes, G.; and Frank, E. 2009. Classifier chains for multi-label classification. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 254–269. Springer.

Read, J.; Pfahringer, B.; Holmes, G.; and Frank, E. 2011. Classifier chains for multi-label classification. *Machine learning*, 85(3): 333–359.

Read, J.; Pfahringer, B.; Holmes, G.; and Frank, E. 2021. Classifier chains: a review and perspectives. *Journal of Artificial Intelligence Research*, 70: 683–718.

Schapire, R. E.; and Singer, Y. 2000. BoosTexter: A boosting-based system for text categorization. *Machine learning*, 39(2): 135–168.

Senge, R.; Coz, J. J. d.; and Hüllermeier, E. 2014. On the problem of error propagation in classifier chains for multi-label classification. In *Data Analysis, Machine Learning and Knowledge Discovery*, 163–170. Springer.

Si, S.; Zhang, H.; Keerthi, S. S.; Mahajan, D.; Dhillon, I. S.; and Hsieh, C.-J. 2017. Gradient boosted decision trees for high dimensional sparse output. In *International conference on machine learning*, 3182–3190. PMLR.

Trajdos, P.; and Kurzynski, M. 2019. Dynamic classifier chains for multi-label learning. In *German Conference on Pattern Recognition*, 567–580. Springer.

Tsoumakas, G.; and Vlahavas, I. 2007. Random k-labelsets: An ensemble method for multilabel classification. In *European conference on machine learning*, 406–417. Springer.

Wang, H.; Liu, W.; Zhao, Y.; Zhang, C.; Hu, T.; and Chen, G. 2019. Discriminative and Correlative Partial Multi-Label Learning. In *IJCAI*, 3691–3697.

Wang, R.; Kwong, S.; Wang, X.; and Jia, Y. 2021. Active k-labelsets ensemble for multi-label classification. *Pattern Recognition*, 109: 107583.

Xia, Y.; Chen, K.; and Yang, Y. 2021. Multi-label classification with weighted classifier selection and stacked ensemble. *Information Sciences*, 557: 421–442.

Zhang, M.; Ramaswamy, H. G.; and Agarwal, S. 2020. Convex calibrated surrogates for the multi-label F-measure. In *International Conference on Machine Learning*, 11246–11255. PMLR.

Zhang, M.-L.; and Zhang, K. 2010. Multi-label learning by exploiting label dependency. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, 999–1008.

Zhang, M.-L.; and Zhou, Z.-H. 2007. ML-KNN: A lazy learning approach to multi-label learning. *Pattern recognition*, 40(7): 2038–2048.

Zhang, Z.; and Jung, C. 2020. GBDT-MO: gradient-boosted decision trees for multiple outputs. *IEEE transactions on neural networks and learning systems*, 32(7): 3156–3167.